EP 501 Final exam

December 11, 2019

This is a take-home exam (work ALL problems) to be turned in 48 hours from the time it is posted (see canvas due date), unless you have notified me of a medical issue, family emergency, or other extenuating circumstance. Unexcused late exams will not be graded.

I will also be checking my email regularly over the next 48 hours to answer questions. If you do not understand a question, please ask for a clarification.

You MAY NOT work in groups or discuss details of the solution to these or similar problems with other students or faculty. You also MAY NOT use online "work this problem for me" resources (i.e. you must complete this problem without direct assistance of another human being). You ARE ALLOWED to use internet references, your notes, textbooks, and other resources such as physics books, differential equations books, integral tables, the TI-89, or mathematical software like Mathematica, Matlab, Maple, and similar.

Include citations, where appropriate, to results that you use for your solutions (e.g. "Jackson eqn. 3.70" "Wolfram website" "Maple software" and so on) and make sure that your work is completely described by your solution (i.e. it adequately developed and explained). Please start a new page for each problem. You must submit your solution via CANVAS as a .zip file with Matlab codes and Word, Pages, or LaTeXtext.

The first two problems are intended to have written solutions and the third problem is intended to have a Matlab script solution. However, you may take any approach you watn to any of these problems. Please submit all codes that you generate for this exam.

You must sign (below) and attach this page to the front of your solution when you submit your solutions for this exam. Electronic signatures are acceptable and encouraged. If you are typing up your solution in LaTeX, please note that the source code for this document is included in the course repository.

I, _____ *, confirm that I did not discuss the solution to these problems with anyone else.*

1. (25 pts) Consider an ODE:

$$\frac{df}{dt} = -\alpha f \qquad (\alpha > 0) \tag{1}$$

differenced in the "non-standard" form:

$$\frac{f^{n+1} - f^n}{\Delta t} = -\alpha \left(\frac{2}{3} f^n + \frac{1}{3} f^{n+1} \right) \tag{2}$$

   (a) Develop an update formula ($f^{n+1} = ...$) for this "non-standard" method

   (b) Derive the conditions under which this method is stable

   (c) Rigorously determine whether this method consistent, viz. whether the numerical solution approaches the exact solution in the limit as $\Delta t \to 0$.

   (d) What is the order of accuracy of the method (e.g. it is $\mathcal{O}(\Delta t^m)$ accurate, you need to find m)?

2. (25 pts) In class we discussed many of the complexities of numerically solving PDEs, particularly that different methods are often needed for different types of equations, and that certain methods can generate artifacts or be unconditionally unstable. Construct a ∼2-4 page, typeset (in Word, Pages, or LaTeX– with equations as appropriate) set of notes that *provides a survey of numerical approaches to PDEs*. Write your notes in narrative form (viz. using complete sentences and proper grammar). Include the following items in your survey:

   - Definitions different PDE types (elliptic, hyperbolic, and parabolic)

     Numerical Approaches to PDEs

     Partial differential equations (PDEs) are equations that involve functions of multiple variables and the partial derivatives of those functions. The three main types of PDEs are elliptic, hyperbolic, and parabolic and they are classified as follows.

     In the case where a PDE takes the form:

$$au_{xx} + bu_{xy} + cu_{yy} = 0, u = u(x, y) \tag{3}$$

     its classification is determined by the value of the following relation:

$$b^2 - 4ac \tag{4}$$

     This function is elliptic if this value is less than 0, hyperbolic if it is greater than 0, and parabolic if it equals 0.

     Elliptic PDEs:

     The canonical form of elliptic PDEs is called the Poisson equation and is as follows:

$$\nabla^2 u = f(x) \tag{5}$$

     when f(x) is 0, this equation becomes the Laplace equation. These equations can be used to model physical phenomenon such as electric potential and gravitational potential. For example, the solution to the equation $\nabla^2 \phi = -\rho/\epsilon_0$ describes the electric potential of the free space surrounding the charge density ρ. The analytical solution of these equations generally take the form of Fourier sine and cosine series and can be obtained through separation of variables.

     In order to solve elliptic equations numerically, a finite difference approach is necessary. By representing the below equation (Laplace equation) with centered, 2nd order accurate finite differences,

a system of equations can be made that will lead to a solution when solved using linear algebra solvers.

$$\frac{\partial^2 \phi}{\partial x^2} + \frac{\partial^2 \phi}{\partial y^2} = 0 \tag{6}$$

let:

$$\left[\frac{\partial^2 \Phi}{\partial x^2}\right]_{i,j} = \frac{\Phi_{i+1,j} - 2\Phi_{i,j} + \Phi_{i-1,j}}{\Delta x^2} \tag{7}$$

and

$$\left[\frac{\partial^2 \Phi}{\partial y^2}\right]_{i,j} = \frac{\Phi_{i,j+1} - 2\Phi_{i,j} + \Phi_{i,j-1}}{\Delta x^2} \tag{8}$$

by substituting these relations into equation (6) and grouping like terms, the following system of equations emerges:

$$\Phi_{i+1,j}\left[\frac{1}{\Delta x^2}\right] + \Phi_{i-1,j}\left[\frac{1}{\Delta x^2}\right] + \Phi_{i,j+1}\left[\frac{1}{\Delta y^2}\right] + \Phi_{i,j-1}\left[\frac{1}{\Delta y^2}\right] + \Phi_{i,j}\left[\frac{-2}{\Delta x^2} + \frac{-2}{\Delta y^2}\right] = 0 \tag{9}$$

In matrix form, this equation becomes

$$\underline{\underline{M}}\underline{\Phi} = \underline{b} \tag{10}$$

where $\underline{\underline{M}}$ is a diagonally dominant matrix containing the various coefficients of $\Phi_{i,j}$. This system can be solved using any linear algebra solver, such as Gaussian elimination or Jacobi iteration. A consideration when performing these operations in MATLAB is the speed in which the program finishes calculating the solution. In order to increase efficiency, the use of sparse storage for $\underline{\underline{M}}$ is recommended.

This matrix only contains the interior coefficients for $\Phi_{i,j}$ and must be separately defined at the boundaries. Typical boundary conditions for elliptic PDEs include Dirichlet and Neumann conditions. Dirichlet boundary conditions are those where the value of the function at the boundaries is defined explicitly. Neumann conditions are those where the value of the derivatives is given at the boundaries. These conditions can be mixed and the value of the derivative can be given at the front boundary while the value of the function is known at the back boundary.

Hyperbolic Equations:
The canonical form of Hyperbolic PDEs is the wave equation, seen below.

$$u_t = -\alpha u_x \tag{11}$$

This equation can be used to model the propagation of a wave through a medium at a specific velocity. An example of this is the propagation of an electromagnetic wave. In this case the hyperbolic equation takes the form

$$\frac{\partial^2 E}{\partial x^2} = \frac{1}{c^2}\frac{\partial^2 E}{\partial t^2}$$

where E is the Electric field and c is the speed of light in the medium in which the wave is propagating. The analytical solution of this type of equations typically results in complex exponential and sinusoidal terms.

In order to solve these equations numerically, an analysis of stability of previous methods must be made. If attempting to use a forward time center space (FTCS) solution, a specific condition must be met. This condition can be derived by checking the growth of a single Fourier mode, $e^{ik_m \Delta x}$. By plugging this mode in for f_i^n, the following condition for the stability is obtained.
The system is stable if

$$\left| 1 - i\frac{v\Delta t}{2\Delta x} sin(k_m \Delta x) \right| \leq 1$$

However, this condition can never be met, and therefor the FTCS method is unconditionally unstable for hyperbolic equations. This is not the case, however, for the forward time, backward space (FTBS) method. A similar analysis can be done using a FTBS update formula (shown below) to achieve a condition for the stability of the system.
FTBS update formula:

$$f_i^{n+1} = f_i^n - \frac{v\Delta t}{\Delta x}\left(f_i^n - f_{i-1}^n\right)$$

Stability condition:

$$\frac{v\Delta t}{\Delta x} \leq 1$$

However, this stability is ruined when $v < 1$. In this case, a forward time, forward space update method can be applied and be stable under the same conditions. By combining these two update formulas, Godunov's method can be obtained. This method employs the concept of upwinding to ensure the solution remains stable. Upwinding involves the use of data "behind" the flow to compute the derivative. This results in the following update formula:

$$f_i^{n+1} = \begin{cases} {}_i^n - \frac{v\Delta t}{\Delta x}\left(f_{i+1}^n - f_i^n\right) \\ v < 0 \\ f_i^n - \frac{v\Delta t}{\Delta x}\left(f_i^n - f_{i-1}^n\right) \\ v > 0 \end{cases}$$

Another conditionally stable method of solving hyperbolic equations is the Lax-Wendroff Method. In order to derive this method, the Lax-Fredrich method must first be discussed. The Lax-Fredrich method employs an FTCS update formula and stabilizes it by introducing an artificial viscosity (averaging of adjacent cells) in the system. This artificial viscosity is a form of numerical diffusion. Numerical diffusion is the tendency for a solution to diffuse in the absence of a true physical viscosity term. This occurs due to truncation errors in finite difference formulas. Although it is a source of error, it allows for the use of FTCS solutions if absolutely necessary. The update formula for the Lax-Fredrich method is as follows:

$$f_i^{n+1} = \frac{1}{2}\left(f_{i+1}^n + f_{i-1}^n\right) - \frac{v\Delta t}{2\Delta x}\left(f_{i+1}^n - f_{i-1}^n\right) \tag{12}$$

This equation was expanded upon to create the Lax-Wendroff method for solving hyperbolic equations. This method employs centered half step updates, both in time and space, to compute the solution. This is beneficial as it allows for a second order accurate solution in space and time for linear advection equations. The update formula is as follows:

$$f_i^{n+1} = f_i^n + \frac{v\Delta t}{2\Delta x}\left(f_{i+1/2}^{n+1/2} - f_{i-1/1}^{n+1/2}\right) \tag{13}$$

where

$$f_{i-1/2}^{n+1/2} = \frac{1}{2}\left(f_i^n + f_{i-1}^n\right) - \frac{v\Delta t}{2\Delta x}\left(f_i^n - f_{i-1}^n\right) \tag{14}$$

and

$$f_{i+1/2}^{n+1/2} = \frac{1}{2}\left(f_{i+1}^{n} + f_{i}^{n}\right) - \frac{v\Delta t}{2\Delta x}\left(f_{i+1}^{n} - f_{i}^{n}\right) \tag{15}$$

Both the Lax-Wendroff method and the Godunov method require initial conditions to begin iterating.

Parabolic Equations:
The canonical form for parabolic equations is of the form

$$u_t - \lambda u_{xx} = 0 \tag{16}$$

This equation is known as the diffusion equation and it describes diffusion though a medium with a diffusivity λ. An example of an equation in this form is the heat equation, seen below. This equation is used to model the diffusion of heat through a medium with thermal diffusivity λ.

$$\frac{\partial T}{\partial t} - \lambda \frac{\partial^2 T}{\partial x^2} = 0 \tag{17}$$

This equation requires initial and boundary conditions to solve and has an analytical solution that includes a combination of exponential and sinusoidal terms. This type of equation can be solved using both explicit and implicit finite difference equations.

Implicit Method:
A FTCS update formula can be developed using a similar method to the methods used previously. Using a forward euler derivative for time and a centered 2nd euler derivative for space, the following update formula can be obtained:

$$T_i^{n+1} = T_i^n + \lambda \frac{\Delta t}{\Delta x}\left(T_{i+1}^n - 2T_i^n + T_{i-2}^n\right) \tag{18}$$

This method is very straight forward and is stable when $\Delta t \leq \frac{\lambda}{2}\Delta x^2$. From this condition, it is clear that fine space grids will require a large amount of small time steps to achieve stability.

Explicit Methods:
In order to obtain an explicit method for solving parabolic PDEs, it is necessary to use backward time, centered space euler differences. After substituting these euler differences into the original formula, the following equation emerges:

$$T_{i-1}^{n+1}\left(\frac{-\lambda}{\Delta x^2}\right) + T_i^{n+1}\left(\frac{1}{\Delta t} + \frac{2\lambda}{\Delta x^2}\right) + T_{i+1}^{n+1}\left(\frac{-\lambda}{\Delta x^2}\right) = \frac{T_i^n}{\Delta t} \tag{19}$$

This can be expressed in matrix notation as $\underline{\underline{M}}\,\underline{T} = \underline{b}$ where $\underline{\underline{M}}$ is a diagonally dominant matrix containing the coefficients of T_i^n, $\underline{T}$ is the vector containing the solution, and $\underline{b}$ is the right hand side of the equation. This system can be solved using any linear algebra solver, such as Jacobi iteration or Gaussian elimination. An extension of this system of equations is the Crank-Nicholson method, also known as the trapezoidal method. This method uses a second order, centered in time finite difference. The resulting system of equations is as follows:

$$T_{i-1}^{n+1}\left(\frac{-\lambda}{2\Delta x^2}\right) + T_i^{n+1}\left(\frac{1}{\Delta t} + \frac{\lambda}{\Delta x^2}\right) + T_{i+1}^{n+1}\left(\frac{-\lambda}{2\Delta x^2}\right) = \frac{T_i^n}{\Delta t} - \frac{\lambda}{2}\frac{T_{i+1}^n - 2T_i^n + T_{i-1}^{n+1}}{\Delta x^2} \tag{20}$$

- An example of a typical physical/engineering system, for each PDE type, that can be modeled with these equations.
- Discuss, in qualitative/intuitive terms, the basic features of the exact solutions of each of these types of equations. You do not need to solve the equations just describe characteristics of the solution (what they "look like").
- The finite difference approach to solving elliptic equations, leading to a system of equations. Give example equations.
- Discuss various options available for solving the large set of FDEs that result.
- Finite difference approaches to solving parabolic equations; discuss the use of explicit vs. implicit methods and give one example of each type of method, including equations and methods of solution (e.g. simple algebraic update formula vs. system of equations).
- Discuss two different conditionally stable methods for solving hyperbolic equations. Also discuss the drawbacks of each of these methods.
- Define numerical diffusion and discuss, conceptually, its role in stabilizing the Lax-Friedrichs method.
- Define and explain the advantage of upwinding. Describe, conceptually, why it is used as the basis for many numerical methods for solving hyperbolic equations.
- Include a list of references for your survey.

3. (50 pts) Explore two different methods to solve the advection-diffusion equation.

$$\frac{\partial f}{\partial t} + \underbrace{v\frac{\partial f}{\partial x}}_{\text{hyperbolic term}} - \underbrace{\lambda\frac{\partial^2 f}{\partial x^2}}_{\text{parabolic term}} = 0 \tag{21}$$

(a) Derive an explicit time update formula for your method using the following differencing approach: (1) a forward Euler time differencing, (2) an upwind difference for the hyperbolic term (describing advection) and (3) a centered difference for the parabolic term (describing diffusion).

(b) Solve this equation for $f(x,t)$ numerically in Matlab using the parameters:

$$v = 20 \tag{22}$$

$$\lambda = 0.25 \tag{23}$$

on the domain $0 \leq x \leq 1$, $0 \leq t \leq 0.05$ with initial condition:

$$f(x,0) = e^{-\frac{(x-0.5)^8}{2(0.1)^8}} \tag{24}$$

and plot your result vs. space and time using `imagesc` or `pcolor, shading flat`.

(c) By repeating your explicit solutions for different choices of Δt, approximate the largest time step that will allow a stable solution. Show an example of your solution for a unstable time step that is just past this stability limit, such that mild instability growth and/or artifacts can be seen.

(d) Change your finite difference method such that the second derivative parabolic term is computed using the solution at the $n+1$ time level, i.e. $f_i^{n+1}, f_{i\pm1}^{n+1}$. Keep the first spatial derivative term as explicit, viz. computed via upwind difference from f_i^n, as it was in part a. This makes the algorithm semi-implicit and means that a system of equations must be solved at each time step. Write down this system and submit it with your solution.

(e) Show that your semi-implicit method from part d produces stable results with much larger time steps than the method derived in part a.

(f) Show that your semi-implicit scheme will eventually become unstable if the time step is too large. It will not look as bad as the fully explicit method in part a, but you can still show that the numerical solution increases in amplitude with time for large time steps - this is unphysical behavior. Plot this instability growth and speculate as to its cause.